

A General Framework for Well-Structured Graph Transformation Systems^{*}

Barbara König and Jan Stückrath

Universität Duisburg-Essen, Germany
 {barbara_koenig, jan.stueckrath}@uni-due.de

Abstract. Graph transformation systems (GTSs) can be seen as well-structured transition systems (WSTSs), thus obtaining decidability results for certain classes of GTSs. In earlier work it was shown that well-structuredness can be obtained using the minor ordering as a well-quasi-order. In this paper we extend this idea to obtain a general framework in which several types of GTSs can be seen as (restricted) WSTSs. We instantiate this framework with the subgraph ordering and the induced subgraph ordering and apply it to analyse a simple access rights management system.

1 Introduction

Well-structured transition systems [2,9] are one of the main sources for decidability results for infinite-state systems. They equip a state space with a quasi-order, which must be a well-quasi-order (wqo) and a simulation relation for the transition relation. If a system can be seen as a WSTS, one can decide the coverability problem, i.e., the problem of verifying whether, from a given initial state, one can reach a state that covers a final state, i.e., is larger than the final state with respect to the chosen order. Often, these given final states, and all larger states, are considered to be error states and one can hence check whether an error state is reachable. Large classes of infinite-state systems are well-structured, for instance (unbounded) Petri nets and certain lossy systems. For these classes of systems the theory provides a generic backwards reachability algorithm.

A natural specification language for concurrent, distributed systems with a variable topology are graph transformation systems [19] and they usually generate infinite state spaces. In those systems states are represented by graphs and state changes by (local) transformation rules, consisting of a left-hand and a right-hand side graph. In [12] it was shown how lossy GTSs with edge contraction rules can be viewed as WSTSs with the graph minor ordering [17,18] and the theory was applied to verify a leader election protocol and a termination detection protocol [4]. The technique works for arbitrary (hyper-)graphs, i.e. the state space is not restricted to certain types of graphs. On the other hand, in order to obtain well-structuredness, we can only allow certain rule sets, for instance one has to require an edge contraction rule for each edge label.

^{*} Research partially supported by DFG project GaReV.

In order to make the framework more flexible we now consider other wqos, different from the minor ordering: the subgraph ordering and the induced subgraph ordering. The subgraph ordering and a corresponding WSTS were already studied in [3], but without the backwards search algorithm. Furthermore, we already mentioned the decidability result in the case of the subgraph ordering in [4], but did not treat it in detail and did not consider it as an instance of a general framework.

In contrast to the minor ordering, the subgraph ordering is not a wqo on the set of all graphs, but only on those graphs where the length of undirected paths is bounded [6]. This results in a trade-off: while the stricter order allows us to consider all possible sets of graph transformation rules in order to obtain a decision procedure, we have to make sure to consider a system where only graphs satisfying this restriction are reachable. Even if this condition is not satisfied, the procedure can yield useful partial coverability results. Also, it often terminates without excluding graphs not satisfying the restriction (this is also the case for our running example), producing exact results. We make these considerations precise by introducing *Q-restricted* WSTSs, where the order need only be a wqo on Q . In general, one wants Q to be as large as possible to obtain stronger statements.

It turns out that the results of [12] can be transferred to this new setting. Apart from the minor ordering and the subgraph ordering, there are various other wqos that could be used [8], leading to different classes of systems and different notions of coverability. In order to avoid redoing the proofs for every case, we here introduce a general framework which works for the case where the partial order can be represented by graph morphisms, which is applicable to several important cases. Especially, we state conditions required to perform the backwards search. We show that the case of the minor ordering can be seen as a special instance of this general framework and show that the subgraph and the induced subgraph orderings are also compatible. Finally we present an implementation and give runtime results. The proofs can be found in the Appendix B.

2 Preliminaries

2.1 Well-structured Transition Systems

We define an extension to the notion of WSTS as introduced in [2,9], a general framework for decidability results for infinite-state systems, based on well-quasi-orders.

Definition 1 (Well-quasi-order and upward closure). *A quasi-order \leq (over a set X) is a well-quasi-order (wqo) if for any infinite sequence x_0, x_1, x_2, \dots of elements of X , there exist indices $i < j$ with $x_i \leq x_j$.*

An upward-closed set is any set $I \subseteq X$ such that $x \leq y$ and $x \in I$ implies $y \in I$. For a subset $Y \subseteq X$, we define its upward closure $\uparrow Y = \{x \in X \mid \exists y \in Y: y \leq x\}$. Then, a basis of an upward-closed set I is a set I_B such that $I = \uparrow I_B$. A downward-closed set, downward closure and a basis of a downward-closed set can be defined analogously.

The definition of wqos gives rise to properties which are important for the correctness and termination of the backwards search algorithm presented later.

Lemma 1. *Let \leq be a wqo, then the following two statements hold:*

1. *Any upward-closed set I has a finite basis.*
2. *For any infinite, increasing sequence of upward-closed sets $I_0 \subseteq I_1 \subseteq I_2 \subseteq \dots$ there exists an index $k \in \mathbb{N}$ such that $I_i = I_{i+1}$ for all $i \geq k$.*

A Q -restricted WSTS is a transition system, equipped with a quasi-order, such that the quasi-order is a (weak) simulation relation on all states and a wqo on a restricted set of states Q .

Definition 2 (Q -restricted well-structured transition system). *Let S be a set of states and let Q be a downward closed subset of S , where membership is decidable. A Q -restricted well-structured transition system (Q -restricted WSTS) is a transition system $\mathcal{T} = (S, \Rightarrow, \leq)$, where the following conditions hold:*

Ordering: \leq is a quasi-order on S and a wqo on Q .

Compatibility: For all $s_1 \leq t_1$ and a transition $s_1 \Rightarrow s_2$, there exists a sequence $t_1 \Rightarrow^* t_2$ of transitions such that $s_2 \leq t_2$.

$$\begin{array}{ccc} t_1 & \xRightarrow{*} & t_2 \\ \vee | & & \vee | \\ s_1 & \xRightarrow{*} & s_2 \end{array}$$

The presented Q -restricted WSTS are a generalization of WSTS and are identical to the classical definition, when $Q = S$. We will show how well-known results for WSTS can be transferred to Q -restricted WSTS. For Q -restricted WSTS there are two coverability problems of interest. The *(general) coverability problem* is to decide, given two states $s, t \in S$, whether there is a sequence of transitions $s \Rightarrow s_1 \Rightarrow \dots \Rightarrow s_n$ such that $t \leq s_n$. The *restricted coverability problem* is to decide whether there is such a sequence for two $s, t \in Q$ with $s_i \in Q$ for $1 \leq i \leq n$. Both problems are undecidable in the general case (as a result of [4] and Proposition 5) but we will show that the well-known backward search for classical WSTS can be put to good use.

Given a set $I \subseteq S$ of states we denote by $Pred(I)$ the set of direct predecessors of I , i.e., $Pred(I) = \{s \in S \mid \exists s' \in I: s \Rightarrow s'\}$. Additionally, we use $Pred_Q(I)$ to denote the restriction $Pred_Q(I) = Pred(I) \cap Q$. Furthermore, we define $Pred^*(I)$ as the set of all predecessors (in S) which can reach some state of I with an arbitrary number of transitions. To obtain decidability results, the sets of predecessors must be computable, i.e. a so-called effective pred-basis must exist.

Definition 3 (Effective pred-basis). *A Q -restricted WSTS has an effective pred-basis if there exists an algorithm accepting any state $s \in S$ and returning $pb(s)$, a finite basis of $\uparrow Pred(\uparrow\{s\})$. It has an effective Q -pred-basis if there exists an algorithm accepting any state $q \in Q$ and returning $pb_Q(q)$, a finite basis of $\uparrow Pred_Q(\uparrow\{q\})$.*

Whenever there exists an effective pred-basis, there also exists an effective Q -pred-basis, since we can use the downward closure of Q to prove $pb_Q(q) = pb(q) \cap Q$.

Let (S, \Rightarrow, \leq) be a Q -restricted WSTS with an effective pred-basis and let $I \subseteq S$ be an upward-closed set of states with finite basis I_B . To solve the general coverability problem we compute the sequence $I_0, I_1, I_2 \dots$ where $I_0 = I_B$ and $I_{n+1} = I_n \cup pb(I_n)$. If the sequence $\uparrow I_0 \subseteq \uparrow I_1 \subseteq \uparrow I_2 \subseteq \dots$ becomes stationary, i.e. there is an m with $\uparrow I_m = \uparrow I_{m+1}$, then $\uparrow I_m = \uparrow Pred^*(I)$ and a state of I is coverable from a state s if and only if there exists an $s' \in I_m$ with $s' \leq s$. If \leq is a wqo on S , by Lemma 1 every upward-closed set is finitely representable and every sequence becomes stationary. However, in general the sequence might not become stationary if $Q \neq S$, in which case the problem becomes semi-decidable, since termination is no longer guaranteed (although correctness is).

The restricted coverability problem can be solved in a similar way, if an effective Q -pred-basis exists. Let $I^Q \subseteq S$ be an upward closed set of states with finite basis $I_B^Q \subseteq Q$. We compute the sequence $I_0^Q, I_1^Q, I_2^Q, \dots$ with $I_0^Q = I_B^Q$ and $I_{n+1}^Q = I_n^Q \cup pb_Q(I_n^Q)$. Contrary to the general coverability problem, the sequence $\uparrow I_0^Q \cap Q \subseteq \uparrow I_1^Q \cap Q \subseteq \uparrow I_2^Q \cap Q \subseteq \dots$ is guaranteed to become stationary according to Lemma 1. Let again m be the first index with $\uparrow I_m^Q = \uparrow I_{m+1}^Q$, and set $\Rightarrow_Q = (\Rightarrow \cap Q \times Q)$. We obtain the following result, of which the classical decidability result of [9] is a special case.

Theorem 1 (Coverability problems). *Let $T = (S, \Rightarrow, \leq)$ be a Q -restricted WSTS with a decidable order \leq .*

- (i) *If T has an effective pred-basis and $S = Q$, the general and restricted coverability problems coincide and both are decidable.*
- (ii) *If T has an effective Q -pred-basis, the restricted coverability problem is decidable if Q is closed under reachability.*
- (iii) *If T has an effective Q -pred-basis and I_m^Q is the limit as described above, then: if $s \in \uparrow I_m^Q$, then s covers a state of I^Q in \Rightarrow (general coverability). If $s \notin \uparrow I_m^Q$, then s does not cover a state of I^Q in \Rightarrow_Q (no restricted coverability).*
- (iv) *If T has an effective pred-basis and the sequence I_n becomes stationary for $n = m$, then: a state s covers a state of I if and only if $s \in \uparrow I_m$.*

Thus, if T is a Q -restricted WSTS and the “error states” can be represented as an upward-closed set I , then the reachability of an error state of I can be determined as described above, depending on which of the cases of Theorem 1 applies. Note that it is not always necessary to compute the limits I_m or I_m^Q , since $\uparrow I_i \subseteq \uparrow I_m$ (and $\uparrow I_i^Q \subseteq \uparrow I_m^Q$) for any $i \in \mathbb{N}_0$. Hence, if $s \in \uparrow I_i$ (or $s \in \uparrow I_i^Q$) for some i , then we already know that s covers a state of I (or of I^Q) in \Rightarrow .

2.2 Graph Transformation Systems

In the following we define the basics of hypergraphs and GTSs as a special form of transition systems where the states are hypergraphs and the rewriting rules

are hypergraph morphisms. We prefer hypergraphs over directed or undirected graphs since they are more convenient for system modelling.

Definition 4 (Hypergraph). Let Λ be a finite sets of edge labels and $ar: \Lambda \rightarrow \mathbb{N}$ a function that assigns an arity to each label. A (Λ) -hypergraph is a tuple (V_G, E_G, c_G, l_G^E) where V_G is a finite set of nodes, E_G is a finite set of edges, $c_G: E_G \rightarrow V_G^*$ is an (ordered) connection function and $l_G^E: E_G \rightarrow \Lambda$ is an edge labelling function. We require that $|c_G(e)| = ar(l_G^E(e))$ for each edge $e \in E_G$.

An edge e is called incident to a node v (and vice versa) if v occurs in $c_G(e)$.

From now on we will often call hypergraphs simply graphs. An (elementary) *undirected path* of length n in a hypergraph is an alternating sequence $v_0, e_1, v_1, \dots, v_{n-1}, e_n, v_n$ of nodes and edges such that for every index $1 \leq i \leq n$ both nodes v_{i-1} and v_i are incident to e_i and the undirected path contains all nodes and edges at most once. Note that there is no established notion of directed paths for hypergraphs, but our definition gives rise to undirected paths in the setting of directed graphs (which are a special form of hypergraphs).

Definition 5 (Partial hypergraph morphism). Let G, G' be (Λ) -hypergraphs. A partial hypergraph morphism (or simply morphism) $\varphi: G \rightarrow G'$ consists of a pair of partial functions $(\varphi_V: V_G \rightarrow V_{G'}, \varphi_E: E_G \rightarrow E_{G'})$ such that for every $e \in E_G$ it holds that $l_G(e) = l_{G'}(\varphi_E(e))$ and $\varphi_V(c_G(e)) = c_{G'}(\varphi_E(e))$ whenever $\varphi_E(e)$ is defined. Furthermore if a morphism is defined on an edge, it must be defined on all nodes incident to it. Total morphisms are denoted by an arrow of the form \rightarrow .

For simplicity we will drop the subscripts and write φ instead of φ_V and φ_E . We call two graphs G_1, G_2 isomorphic if there exists a total bijective morphism $\varphi: G_1 \rightarrow G_2$.

Graph rewriting relies on the notion of *pushouts*. It is known that pushouts of partial graph morphisms always exist and are unique up to isomorphism. Intuitively, for morphisms $\varphi: G_0 \rightarrow G_1, \psi: G_0 \rightarrow G_2$, the pushout is obtained by gluing the two graphs G_1, G_2 over the common interface G_0 and by deleting all elements which are undefined under φ or ψ (for a formal definition see Appendix A).

We will take pushouts mainly in the situation described in Definition 6 below, where r (the rule) is partial and connects the left-hand side L and the right-hand side R . It is applied to a graph G via a total match m . In order to ensure that the resulting morphism m' (the co-match of the right-hand side in the resulting graph) is also total, we have to require a match m to be *conflict-free* wrt. r , i.e., if there are two elements x, y of L with $m(x) = m(y)$ either $r(x), r(y)$ are both defined or both undefined. Here we consider a graph rewriting approach called the *single-pushout approach (SPO)* [7], since it relies on one pushout square, and restrict to conflict-free matches.

Definition 6 (Graph rewriting). A rewriting rule is a partial morphism $r: L \rightarrow R$, where L is called left-hand and R right-hand side. A match (of r) is

a total morphism $m: L \rightarrow G$, conflict-free wrt. r . Given a rule and a match, a rewriting step or rule application is given by a pushout diagram as shown below, resulting in the graph H .

A graph transformation system (GTS) is a finite set of rules \mathcal{R} .

Given a fixed set of graphs \mathcal{G} , a graph transition system on \mathcal{G} generated by a graph transformation system \mathcal{R} is represented by a tuple $(\mathcal{G}, \Rightarrow)$ where \mathcal{G} is the set of states and $G \Rightarrow G'$ if and only if $G, G' \in \mathcal{G}$ and G can be rewritten to G' using a rule of \mathcal{R} .

$$\begin{array}{ccc} L & \xrightarrow{r} & R \\ m \downarrow & & \downarrow m' \\ G & \longrightarrow & H \end{array}$$

Later we will have to apply rules backwards, which means that it is necessary to compute so-called pushout complements, i.e., given r and m' above, we want to obtain G (such that m is total and conflict-free). How this computation can be performed in general is described in [10]. Note that pushout complements are not unique and possibly do not exist for arbitrary morphisms. For two partial morphisms the number of pushout complements may be infinite.

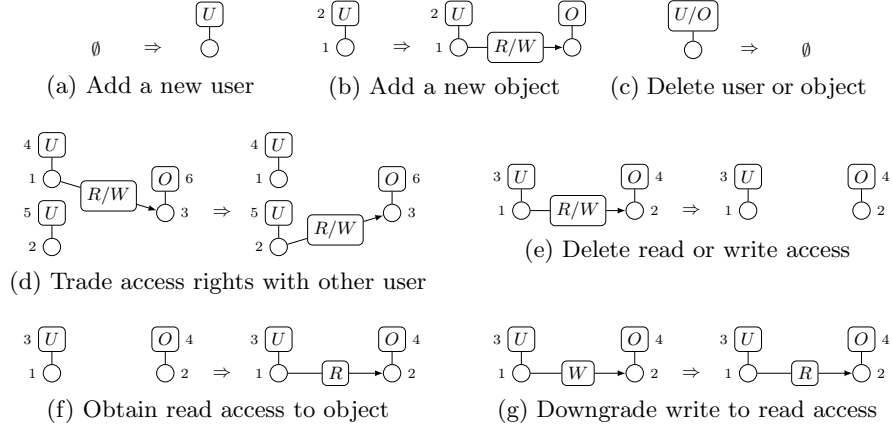


Fig. 1: A GTS modelling a multi-user system

Example 1. To illustrate graph rewriting we model a multi-user system as a GTS (see Figure 1) inspired by [14]. A graph contains user nodes, indicated by unary U -edges, and object nodes, indicated by unary O -edges. Users can have read (R) or write (W) access rights regarding objects indicated by a (directed) edge. Note that binary edges are depicted by arrows, the numbers describe the rule morphisms and labels of the form R/W represent two rules, one with R -edges and one with W -edges.

The users and objects can be manipulated by rules for adding new users (Fig. 1a), adding new objects with read or write access associated with a user (Fig. 1b) and deleting users or objects (Fig. 1c). Both read and write access can

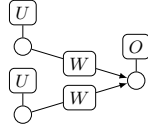


Fig. 2: An undesired state in the multi-user system

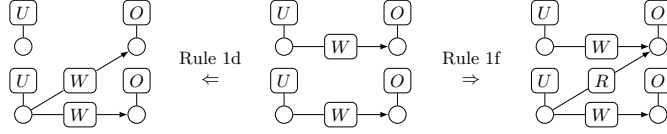


Fig. 3: Example of two rule applications

be traded between users (Fig. 1d) or dropped (Fig. 1e). Additionally users can downgrade their write access to a read access (Fig. 1g) and obtain read access of arbitrary objects (Fig. 1f).

In a multi-user system there can be arbitrary many users with read access to an object, but at most one user may have write access. This means especially that any configuration of the system containing the graph depicted in Figure 2 is erroneous.

An application of the Rules 1d and 1f is shown in Figure 3. In general, nodes and edges on which the rule morphism r is undefined are deleted and nodes and edges of the right-hand side are added if they have no preimage under r . In the case of non-injective rule morphisms, nodes or edges with the same image are merged. Finally, node deletion results in the deletion of all incident edges (which would otherwise be left dangling). For instance, if Rule 1c is applied, all read/write access edges attached to the single deleted node will be deleted as well.

3 GTS as WSTS: A General Framework

In this section we state some sufficient conditions such that the coverability problems for \mathcal{Q} -restricted well-structured GTS can be solved in the sense of Theorem 1 (in the following we use \mathcal{Q} to emphasize that \mathcal{Q} is a set of graphs). We will also give an appropriate backward algorithm. The basic idea is to represent the wqo by a given class of morphisms.

Definition 7 (Representable by morphisms). *Let \sqsubseteq be a quasi-order that satisfies $G_1 \sqsubseteq G_2$, $G_2 \sqsubseteq G_1$ for two graphs G_1, G_2 if and only if G_1, G_2 are isomorphic, i.e., \sqsubseteq is anti-symmetric up to isomorphism.*

We call \sqsubseteq representable by morphisms if there is a class of (partial) morphisms $\mathcal{M}_{\sqsubseteq}$ such that for two graphs G, G' it holds that $G' \sqsubseteq G$ if and only if there is a morphism $(\mu: G \mapsto G') \in \mathcal{M}_{\sqsubseteq}$. Furthermore, for $(\mu_1: G_1 \mapsto G_2), (\mu_2: G_2 \mapsto G_3) \in \mathcal{M}_{\sqsubseteq}$ it holds that $\mu_2 \circ \mu_1 \in \mathcal{M}_{\sqsubseteq}$, i.e., $\mathcal{M}_{\sqsubseteq}$ is closed under composition. We call such morphisms μ order morphisms.

The intuition behind an order morphism is the following: whenever there is an order morphism from G to G' , we usually assume that G' is the smaller graph that can be obtained from G by some form of node deletion, edge deletion or

edge contraction. For any graphs G (which represent all larger graphs) we can now compose rules and order morphisms to simulate a co-match of a rule to some graph larger than G . However, for this construction to yield correct results, the order morphisms have to satisfy the following two properties.

Definition 8 (Pushout preservation). *We say that a set of order morphisms $\mathcal{M}_{\sqsubseteq}$ is preserved by total pushouts if the following holds: if $(\mu: G_0 \mapsto G_1) \in \mathcal{M}_{\sqsubseteq}$ is an order morphism and $g: G_0 \rightarrow G_2$ is total, then the morphism μ' in the pushout diagram on the right is an order morphism of $\mathcal{M}_{\sqsubseteq}$.*

$$\begin{array}{ccc} G_0 & \xrightarrow{\mu} & G_1 \\ \downarrow g & & \downarrow g' \\ G_2 & \xrightarrow{\mu'} & G_3 \end{array}$$

The next property is needed to ensure that every graph G , which is rewritten to a graph H larger than S , is represented by a graph G' obtained by a backward rewriting step from S , i.e. the backward step need not be applied to H .

Definition 9 (Pushout closure). *Let $m: L \rightarrow G$ be total and conflict-free wrt. $r: L \rightarrow R$. A set of order morphisms is called pushout closed if the following holds: if the diagram below on the left is a pushout and $\mu: H \mapsto S$ an order morphism, then there exist graphs R' and G' and order morphisms $\mu_R: R \mapsto R'$, $\mu_G: G \mapsto G'$, such that:*

1. *the diagram below on the right commutes and the outer square is a pushout.*
2. *the morphisms $\mu_G \circ m: L \rightarrow G'$ and $n: R' \rightarrow S$ are total and $\mu_G \circ m$ is conflict-free wrt. r .*

$$\begin{array}{ccc} L & \xrightarrow{r} & R \\ m \downarrow & & \downarrow m' \\ G & \xrightarrow{r'} & H \\ & \searrow \mu & \\ & & S \end{array} \qquad \begin{array}{ccc} L & \xrightarrow{r} & R \xrightarrow{\mu_R} R' \\ m \downarrow & & \downarrow m' \quad \downarrow n \\ G & \xrightarrow{r'} & H \\ \mu_G \downarrow & & \searrow \mu \\ G' & \xrightarrow{s} & S \end{array}$$

We now present a generic backward algorithm for (partially) solving both coverability problems. The procedure has two variants, which both require a GTS, an order and a set of final graphs to generate a set of minimal representatives of graphs covering a final graph. The first variant computes the sequence I_n^Q and restricts the set of graphs to ensure termination. It can be used for cases (i), (ii) and (iii) of Theorem 1, while the second variant computes I_n (without restriction) and can be used for cases (i) and (iv).

Procedure 1 (Computation of the (Q) -pred-basis).

Input: A set \mathcal{R} of graph transformation rules, a quasi-order \sqsubseteq on all graphs which is a wqo on a downward-closed set \mathcal{Q} and a finite set of final graphs \mathcal{F} , satisfying:

- The transition system generated by the rule set \mathcal{R} is a \mathcal{Q} -restricted WSTS with respect to the order \sqsubseteq .

- The order \sqsubseteq is representable by a class of morphisms $\mathcal{M}_{\sqsubseteq}$ (Definition 7) and this class satisfies Definitions 8 and 9.
- *Variant 1.* The set of minimal pushout complements restricted to \mathcal{Q} with respect to \sqsubseteq is computable, for all pairs of rules and co-matches (it is automatically finite).
- *Variant 2.* The set of minimal pushout complements with respect to \sqsubseteq is finite and computable, for all pairs of rules and co-matches.

Preparation: Generate a new rule set \mathcal{R}' from \mathcal{R} in the following way: for every rule $(r : L \rightarrow R) \in \mathcal{R}$ and every order morphism $\mu : R \rightarrow \overline{R}$ add the rule $\mu \circ r$ to \mathcal{R}' . (Note that it is sufficient to take a representative \overline{R} for each of the finitely many isomorphism classes, resulting in a finite set \mathcal{R}' .) Start with the working set $\mathcal{W} = \mathcal{F}$ and apply the first backward step.

Backward Step: Perform backward steps until the sequence of working sets \mathcal{W} becomes stationary. The following substeps are performed in one backward step for each rule $(r : L \rightarrow R) \in \mathcal{R}'$:

1. For a graph $G \in \mathcal{W}$ compute all total morphisms $m' : R \rightarrow G$ (co-matches of R in G).
2. *Variant 1.* For each such morphism m' calculate the set \mathcal{G}_{poc} of minimal pushout complement objects of m' with rule r , which are also elements of \mathcal{Q} .
Variant 2. Same as Variant 1, but calculate *all* minimal pushout complements, without the restriction to \mathcal{Q} .
3. Add all remaining graphs in \mathcal{G}_{poc} to \mathcal{W} and minimize \mathcal{W} by removing all graphs G' for which there is a graph $G'' \in \mathcal{W}$ with $G' \neq G''$ and $G'' \sqsubseteq G'$.

Result: The resulting set \mathcal{W} contains minimal representatives of graphs from which a final state is coverable (cf. Theorem 1).

The reason for composing rule morphisms with order morphisms when doing the backwards step is the following: the graph G , for which we perform the step, might not contain a right-hand side R in its entirety. However, G can represent graphs that do contain R and hence we have to compute the effect of applying the rule backwards to all graphs represented by G . Instead of enumerating all these graphs (which are infinitely many), we simulate this effect by looking for matches of right-hand sides modulo order morphisms. We show that the procedure is correct by proving the following lemma.

Proposition 1. *Let $pb_1()$ and $pb_2()$ be a single backward step of Procedure 1 for Variant 1 and 2 respectively. For each graph S , $pb_1(S)$ is an effective \mathcal{Q} -pred-basis and $pb_2(S)$ is an effective pred-basis.*

4 Well-quasi Orders for Graph Transformation Systems

4.1 Minor Ordering

We first instantiate the general framework with the minor ordering, which was already considered in [12]. The minor ordering is a well-known order on graphs,

which is defined as follows: a graph G is a minor of G' whenever G can be obtained from G' by a series of node deletions, edge deletions and edge contractions, i.e. deleting an edge and merging its incident nodes according to an arbitrary partition. Robertson and Seymour showed in a seminal result that the minor ordering is a wqo on the set of all graphs [17], even for hypergraphs [18], thus case (i) of Theorem 1 applies. In [12,13] we showed that the conditions for WSTS are satisfied for a restricted set of GTS by introducing minor morphisms and proving a result analogous to Proposition 1, but only for this specific case. The resulting algorithm is a special case of both variants of Procedure 1.

Proposition 2 ([12]). *The coverability problem is decidable for every GTS if the minor ordering is used and the rule set contains edge contraction rules for each edge label.*

4.2 Subgraph Ordering

In this paper we will show that the subgraph ordering and the induced subgraph ordering satisfy the conditions of Procedure 1 for a restricted set of graphs and are therefore also compatible with our framework. For the subgraph ordering we already stated a related result (but for injective instead of conflict-free matches) in [4], but did not yet instantiate a general framework.

Definition 10 (Subgraph). *Let G_1, G_2 be graphs. G_1 is a subgraph of G_2 (written $G_1 \subseteq G_2$) if G_1 can be obtained from G_2 by a sequence of deletions of edges and isolated nodes. We call a partial morphism $\mu: G \rightarrow S$ a subgraph morphism if and only if it is injective on all elements on which it is defined and surjective.*

It can be shown that the subgraph ordering is representable by subgraph morphisms, which satisfy the necessary properties. Using a result from Ding [6] we can show that the set \mathcal{G}_k of hypergraphs where the length of every undirected path is bounded by k , is well-quasi-ordered by the subgraph relation. A similar result was shown by Meyer for depth-bounded systems in [16]. Note that we bound undirected path lengths instead of directed path lengths. For the class of graphs with bounded directed paths there exists a sequence of graphs violating the wqo property (a sequence of circles of increasing length, where the edge directions alternate along the circle).

Since every GTS satisfies the compatibility condition of Definition 2 naturally, we obtain the following result.

Proposition 3 (WSTS wrt. the subgraph ordering). *Let k be a natural number. Every graph transformation system forms a \mathcal{G}_k -restricted WSTS with the subgraph ordering.*

The set of minimal pushout complements (not just restricted to \mathcal{G}_k) is always finite and can be computed as in the minor case.

Proposition 4. *Every \mathcal{G}_k -restricted well-structured GTS with the subgraph order has an effective pred-basis and the (decidability) results of Theorem 1 apply.*

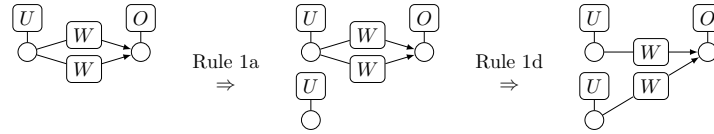
By a simple reduction from the reachability problem for two counter machines, we can show that the restricted coverability problem is undecidable in the general case. Although we cannot directly simulate the zero test, i.e. negative application conditions are not possible, we can make sure that the rules simulating the zero test are applied correctly if and only if the bound k was not exceeded.

Proposition 5. *Let $k > 2$ be a natural number. The restricted coverability problem for \mathcal{G}_k -restricted well-structured GTS with the subgraph ordering is undecidable.*

Example 2. Now assume that an error graph is given and that a graph exhibits an error if and only if it contains the error graph as a subgraph. Then we can use Proposition 4 to calculate all graphs which lead to some error configuration.

For instance, let a multi-user system as described in Example 1 be given. Normally we have to choose a bound on the undirected path length to guarantee termination, but in this example Variant 2 of Procedure 1 terminates and we can solve coverability on the unrestricted transition system (see Theorem 1(iv)). The graph in Figure 2 represents the error in the system and by applying Procedure 1 we obtain a set of four graphs (one of which is the error graph itself), fully characterizing all predecessor graphs. We can observe that the error can only be reached from graphs already containing two W -edges going to a single object node. Hence, the error is not produced by the given rule set if we start with the empty graph and thus the system is correct.

Interestingly the backward search finds the leftmost graph below due to the depicted sequence of rule applications, which leads directly to the error graph. Thus, the error can occur even if a single user has two write access rights to an object, because of access right trading.



The other two graphs computed are shown below and represent states with "broken" structure (a node cannot be a user *and* an object). The left graph for instance can be rewritten to a graph larger than the left graph above, by a non-injective match of the rule in Figure 1d mapping both nodes 2 and 3 to the right node.



4.3 Induced Subgraph Ordering

As for the subgraph ordering in Section 4.2 our backward algorithm can also be applied to the induced subgraph ordering, where a graph G is considered as an induced subgraph of G' if every edge in G' connecting only nodes also present in G , is contained in G as well. Unfortunately, this ordering is not a wqo even when bounding the longest undirected path in a graph, such that we also have to bound the multiplicity of edges between two nodes. Note that this restriction is implicitly done in [6] since Ding uses simple graphs.

Furthermore, since we do not know whether the induced subgraph ordering can be extended to a wqo on (a class of) hypergraphs, we here use only *directed graphs*, where each edge is connected to a sequence of exactly two nodes. For many applications directed graphs are sufficient for modelling, also for our examples, since unary hyperedges can simply be represented by loops.

At first, this order seems unnecessary, since it is stricter than the subgraph ordering and is a wqo on a more restricted set of graphs. On the other hand, it allows us to specify error graphs more precisely, since a graph $G \in \mathcal{F}$ does not represent graphs with additional edges between nodes of G . Furthermore one could equip the rules with a limited form of negative application conditions, still retaining the compatibility condition of Definition 2.

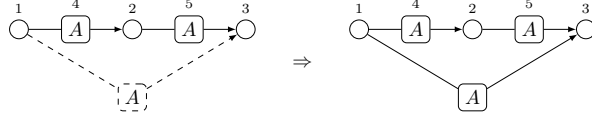
Definition 11 (Induced subgraph). *Let G_1, G_2 be graphs. G_1 is an induced subgraph of G_2 (written $G_1 \trianglelefteq G_2$) if G_1 can be obtained from G_2 by deleting a subset of the nodes and all incident edges. We call a partial morphism $\mu: G \rhd \rightarrow S$ an induced subgraph morphism if and only if it is injective for all elements on which is defined, surjective, and if it is undefined on an edge e , it is undefined on at least one node incident to e .*

Proposition 6 (WSTS wrt. the induced subgraph ordering). *Let n, k be natural numbers and let $\mathcal{G}_{n,k}$ be a set of directed, edge-labelled graphs, where the longest undirected path is bounded by n and every two nodes are connected by at most k parallel edges with the same label (bounded edge multiplicity). Every GTS forms a $\mathcal{G}_{n,k}$ -restricted WSTS with the induced subgraph ordering.*

Proposition 7. *Every $\mathcal{G}_{n,k}$ -restricted well-structured GTS with the induced subgraph order has an effective $\mathcal{G}_{n,k}$ -pred-basis and the (decidability) results of Theorem 1 apply.*

The computation of minimal pushout complements in this case is considerably more complex, since extra edges have to be added (see the proof of Proposition 7 in Appendix B), but we also obtain additional expressiveness. In general GTS with negative application conditions do not satisfy the compatibility condition with respect to the subgraph relation, but we show in the following example, that it may still be satisfied with respect to the induced subgraph relation.

Example 3. Let the following simple rule be given, where the negative application condition is indicated by the dashed edge, i.e. the rule is applicable if and only if there is a matching only for the solid part of the left-hand side and this matching cannot be extended to match also the dashed part.



Applied to a graph containing only A -edges, this rule calculates the transitive closure and will terminate at some point. This GTS satisfies the compatibility condition wrt. the induced subgraph ordering, since for instance a directed path of length two (the left-hand side) does not represent graphs where there is an edge from the first to the last node of the graph. Therefore we can use the induced subgraph ordering and our procedure to show that a graph containing two parallel A -edges can only be reached from graphs already containing two parallel A -edges.

The principle described in the example can be extended to all negative application conditions which forbid the existence of edges but not of nodes. This is the case, because if there is no edge between two nodes of a graph, there is also no edge between these two nodes in any larger graph. Hence if there is no mapping from the negative application condition into the smaller graph, there can also be none into the larger graph. Graphs violating the negative application condition are simply not represented by the smaller graph. Hence, all graph transformation rules with such negative application conditions satisfy the compatibility condition wrt. the induced subgraph ordering. The backward step has to be modified in this case by dropping all obtained graphs which do not satisfy one of the negative application conditions.

4.4 Implementation

We implemented Procedure 1 with support for the minor ordering as well as the subgraph ordering in the tool UNCOVER. The tool is written in C++ and designed in a modular way for easy extension with further orders. The sole optimization currently implemented is the omission of all rules that are also order morphisms. It can be shown that the backward application of such rules produces only graphs which are already represented.

Table 1 shows the runtime results of different case studies, namely a leader election protocol and a termination detection protocol (in an incorrect as well as a correct version), using the minor ordering, and the access rights management protocol described in Figure 1 as well as a public-private server protocol, using the subgraph order. It shows for each case the restricted graph set \mathcal{Q} , the variant of the procedure used (for the minor ordering they coincide), the runtime and the number of minimal graphs representing all predecessors of error graphs.

5 Conclusion

We have presented a general framework for viewing GTSs as restricted WSTSs. We showed that the work in [12] for the minor ordering can be seen as an instance

Table 1: Runtime result for different case studies

case study	wqo	graph set \mathcal{Q}	variant	time	#(error graphs)
Leader election	minor	all graphs	1 / 2	3s	38
Termination detection (faulty)	minor	all graphs	1 / 2	7s	69
Termination detection (correct)	minor	all graphs	1 / 2	2s	101
Rights management	subgraph	all graphs	2	1s	4
Public-private server ($l = 5$)	subgraph	path ≤ 5	1	1s	14
Public-private server ($l = 6$)	subgraph	path ≤ 6	1	16s	16

of this framework and we presented two additional instantiations, based on the subgraph ordering and the induced subgraph ordering. Furthermore we presented the management of read and write access rights as an example and discussed our implementation with very encouraging runtime results.

Currently we are working on an extension of the presented framework with rules, which can uniformly change the entire neighbourhood of nodes. In this case the computed set of predecessor graphs will be an over-approximation. More extensions are possible (possibly introducing over-approximations) and we especially plan to further investigate the integration of rules with negative application conditions as for the induced subgraph ordering. In [15] we introduced an extension with negative application conditions for the minor ordering, but still, the interplay of the well-quasi-order and conditions has to be better understood. Naturally, we plan to look for additional orders, for instance the induced minor and topological minor orderings [8] in order to see whether they can be integrated into this framework and to study application scenarios.

Related work. Related to our work is [3], where the authors use the subgraph ordering and a forward search to prove fair termination for depth-bounded systems. In [1] another wqo for well-structuring graph rewriting is considered, however only for graphs where every node has out-degree 1. It would be interesting to see whether this wqo can be integrated into our general framework. The work in [5] uses the induced subgraph ordering to verify broadcast protocols. There the rules are different from our setting: a left-hand side consists of a node and its entire neighbourhood of arbitrary size. Finally [20] uses a backwards search on graph patterns in order to verify an ad-hoc routing protocol, but not in the setting of WSTSs.

Acknowledgements: We would like to thank Roland Meyer, for giving us the idea to consider the subgraph ordering on graphs, and Giorgio Delzanno for several interesting discussions on wqos and WSTSs.

References

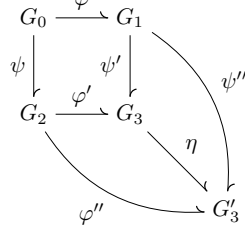
1. P. Aziz Abdulla, A. Bouajjani, J. Cederberg, F. Haziza, and A. Rezine. Monotonic abstraction for programs with dynamic memory heaps. In *Proc. of CAV '08*, pages 341–354. Springer, 2008. LNCS 5123.

2. P. Aziz Abdulla, K. Čerāns, B. Jonsson, and Y. Tsay. General decidability theorems for infinite-state systems. In *Proc. of LICS '96*, pages 313–321. IEEE, 1996.
3. K. Bansal, E. Koskinen, T. Wies, and D. Zufferey. Structural counter abstraction. In *Proc. of TACAS '13*, TACAS'13, pages 62–77. Springer, 2013.
4. N. Bertrand, G. Delzanno, B. König, A. Sangnier, and J. Stückrath. On the decidability status of reachability and coverability in graph transformation systems. In *Proc. of RTA '12*, volume 15 of *LIPICs*, pages 101–116. Schloss Dagstuhl – Leibniz Center for Informatics, 2012.
5. G. Delzanno, A. Sangnier, and G. Zavattaro. Parameterized verification of ad hoc networks. In *Proc. CONCUR '10*, pages 313–327. Springer, 2010. LNCS 6269.
6. G. Ding. Subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 16:489–502, November 1992.
7. H. Ehrig, R. Heckel, M. Korff, M. Löwe, L. Ribeiro, A. Wagner, and A. Corradini. Algebraic approaches to graph transformation—part II: Single pushout approach and comparison with double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.1: Foundations*, chapter 4. World Scientific, 1997.
8. M.R. Fellows, D. Hermelin, and F.A. Rosamond. Well-quasi-orders in subclasses of bounded treewidth graphs. In *Proc. of IWPEC '09 (Parameterized and Exact Computation)*, pages 149–160. Springer, 2009. LNCS 5917.
9. A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256(1-2):63–92, April 2001.
10. M. Heumüller, S. Joshi, B. König, and J. Stückrath. Construction of pushout complements in the category of hypergraphs. In *Proc. of GCM '10 (Workshop on Graph Computation Models)*, 2010.
11. G. Higman. Ordering by divisibility in abstract algebras. *Proc. London Math. Soc.*, s3-2(1):326–336, January 1952.
12. S. Joshi and B. König. Applying the graph minor theorem to the verification of graph transformation systems. In *Proc. of CAV '08*, pages 214–226. Springer, 2008. LNCS 5123.
13. S. Joshi and B. König. Applying the graph minor theorem to the verification of graph transformation systems. Technical Report 2012-01, Abteilung für Informatik und Angewandte Kognitionswissenschaft, Universität Duisburg-Essen, 2012.
14. M. Koch, L.V. Mancini, and F. Parisi-Presicce. Decidability of safety in graph-based models for access control. In *Proceedings of the 7th European Symposium on Research in Computer Security*, pages 229–243. Springer, 2002. LNCS 2502.
15. B. König and J. Stückrath. Well-structured graph transformation systems with negative application conditions. In *Proc. of ICGT '12*, pages 89–95. Springer, 2012. LNCS 7562.
16. R. Meyer. *Structural Stationarity in the π -Calculus*. PhD thesis, Carl-von-Ossietzky-Universität Oldenburg, 2009.
17. N. Robertson and P. Seymour. Graph minors. XX. Wagner’s conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.
18. N. Robertson and P. Seymour. Graph minors XXIII. Nash-Williams’ immersion conjecture. *Journal of Combinatorial Theory Series B*, 100:181–205, March 2010.
19. G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.1: Foundations*. World Scientific, 1997.
20. M. Saksena, O. Wibling, and B. Jonsson. Graph grammar modeling and verification of ad hoc routing protocols. In *Proc. of TACAS '08*, pages 18–32. Springer, 2008. LNCS 4963.

A Pushouts

We give the definition and construction of pushouts, the graph gluing construction which is used in this paper.

Definition 12. Let $\varphi: G_0 \rightarrow G_1$ and $\psi: G_0 \rightarrow G_2$ be two partial graph morphisms. The pushout of φ and ψ consists of a graph G_3 and two morphisms $\psi': G_1 \rightarrow G_3$, $\varphi': G_2 \rightarrow G_3$ such that $\psi' \circ \varphi = \varphi' \circ \psi$ and for every other pair of morphisms $\psi'': G_1 \rightarrow G'_3$, $\varphi'': G_2 \rightarrow G'_3$ such that $\psi'' \circ \varphi = \varphi'' \circ \psi$ there exists a unique morphism $\eta: G_3 \rightarrow G'_3$ with $\eta \circ \psi' = \psi''$ and $\eta \circ \varphi' = \varphi''$.



Proposition 8 (Construction of pushouts). Let $\varphi: G_0 \rightarrow G_1$, $\psi: G_0 \rightarrow G_2$ be partial hypergraph morphisms. Furthermore let \equiv_V be the smallest equivalence on $V_{G_1} \cup V_{G_2}$ and \equiv_E the smallest equivalence on $E_{G_1} \cup E_{G_2}$ such that $\varphi(x) \equiv \psi(x)$ for every element x of G_0 .

An equivalence class of nodes is called valid if it does not contain the image of a node x of G_0 for which $\varphi(x)$ or $\psi(x)$ are undefined. Similarly a class of edges is valid if the analogous condition holds and furthermore all nodes incident to these edges are contained in valid equivalence classes.

Then the pushout graph G_3 of φ and ψ consists of all valid equivalence classes $[x]_{\equiv}$ of nodes and edges, where $l_{G_3}([e]_{\equiv}) = l_{G_i}(e)$ and $c_{G_3}([e]_{\equiv}) = [v_1]_{\equiv} \dots [v_k]_{\equiv}$ if $e \in E_{G_i}$ and $c_{G_i}(e) = v_1 \dots v_k$. Furthermore the morphisms ψ', φ' map nodes and edges to their respective equivalence classes.

Definition 13 (Pushout complement). Let $\varphi: G_0 \rightarrow G_1$ and $\psi': G_1 \rightarrow G_3$ be morphisms. We call the graph G_2 together with the morphisms $\psi: G_0 \rightarrow G_2$ and $\varphi': G_2 \rightarrow G_3$ a pushout complement, if G_3, φ', ψ' is the pushout of φ and ψ .

B Proofs

B.1 Well-structured Transition Systems

Theorem 1 (Coverability problems). Let $T = (S, \Rightarrow, \leq)$ be a Q -restricted WSTS with a decidable order \leq .

- (i) If T has an effective pred-basis and $S = Q$, the general and restricted coverability problems coincide and both are decidable.
- (ii) If T has an effective Q -pred-basis, the restricted coverability problem is decidable if Q is closed under reachability.
- (iii) If T has an effective Q -pred-basis and I_m^Q is the limit as described above, then: if $s \in \uparrow I_m^Q$, then s covers a state of I^Q in \Rightarrow (general coverability). If $s \notin \uparrow I_m^Q$, then s does not cover a state of I^Q in \Rightarrow_Q (no restricted coverability).

(iv) If T has an effective pred-basis and the sequence I_n becomes stationary for $n = m$, then: a state s covers a state of I if and only if $s \in \uparrow I_m$.

Proof. (i) is just a reformulation of the decidability results for WSTS. Similar for (ii), if Q is closed under reachability, a Q -restricted WSTS can be seen as a WSTS with state space Q .

We now consider (iii) where Q is not required to be closed under reachability. Assume that $s \in \uparrow I_m^Q$, where I_m^Q has been obtained by fixed-point iteration, starting with the upward-closed set I_B^Q and computing the sequence I_n^Q . By induction we show the existence of a sequence of transitions leading from s to some state in $\uparrow I_B^Q$. Obviously there is an $q_m \in I_m^Q$ with $q_m \leq s$ and by definition either $q_m \in I_{m-1}^Q$ or there are $q_{m-1} \in I_{m-1}^Q$ and q'_{m-1} with $q_m \Rightarrow q'_{m-1}$ and $q_{m-1} \leq q'_{m-1}$. In the latter case, because of the compatibility condition of Definition 2, there is a q''_{m-1} with $s \Rightarrow^* q''_{m-1}$ and $q_{m-1} \leq q'_{m-1} \leq q''_{m-1}$, i.e. s can reach an element of $\uparrow I_{m-1}^Q$. Since this argument holds for q''_{m-1} as well, the state s can ultimately reach a state $q''_0 \in \uparrow I_B^Q$. Note that it is possible that $s = q''_0$, but it is not guaranteed that $q''_n \in Q$ for every n .

For the other statement assume that $s \notin \uparrow I_m^Q$ and assume that there exists a path $s = q_0 \Rightarrow_Q q_1 \Rightarrow_Q \dots \Rightarrow_Q q_k \in \uparrow I_B^Q$. Note that the second assumption is trivially false, if $s \notin Q$. We can show by induction and by definition of $pb_Q()$ that $q_i \in \uparrow I_{k-i}^Q$ and hence $q_0 \in \uparrow I_k^Q \subseteq \uparrow I_m^Q$, which leads to a contradiction.

The proof of case (iv) is straightforward by observing that the set I_m is an exact representation of all predecessors of I . \square

B.2 GTS as WSTS: A General Framework

Lemma 2. *The sets generated by $pb_1(S)$ and $pb_2(S)$ are both finite subsets of $Pred(\uparrow\{S\})$ and $pb_1(S) \subseteq \mathcal{Q}$.*

Proof. By the conditions of Procedure 1, the sets of minimal pushout complements – in the case of $pb_1(S)$ restricted to \mathcal{Q} – are finite and computable. Since the set of rules is also finite, $pb_1(S)$ and $pb_2(S)$ are finite as well. Every non-minimal pushout complement in \mathcal{Q} is represented by a minimal pushout complement in \mathcal{Q} , because of the downward closure of \mathcal{Q} . Thus, $pb_1(S) \subseteq \mathcal{Q}$ holds.

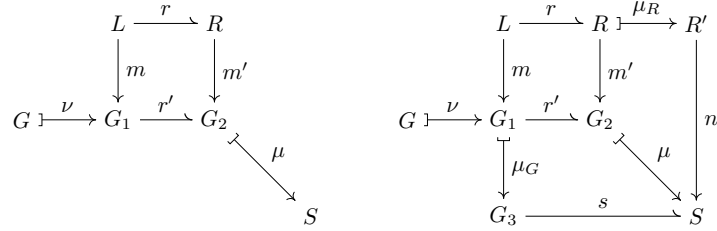
Let $G \in pb_1(S) \cup pb_2(S)$ be a graph generated by one of the procedures. Then there is a rule $r: L \rightarrow R$, an order morphism $\mu: R \rightarrow R'$ and a conflict-free match $m: L \rightarrow G$, such that the left diagram below is a pushout.

$$\begin{array}{ccc}
 L & \xrightarrow{\mu \circ r} & R' \\
 \downarrow m & & \downarrow m'' \\
 G & \xrightarrow{k} & S
 \end{array}
 \qquad
 \begin{array}{ccccc}
 L & \xrightarrow{r} & R & \xrightarrow{\mu} & R' \\
 \downarrow m & & \downarrow m' & & \downarrow m'' \\
 G & \xrightarrow{r'} & S' & \xrightarrow{\mu'} & S \\
 & \searrow k & & &
 \end{array}$$

Let $m': R \rightarrow S'$, $r': G \rightarrow S'$ be the pushout of m , r . Because the outer diagram on the right commutes, there is a unique morphism $\mu': S' \rightarrow S$. The left and the outer square are both pushouts and therefore also the right square is a pushout. Since m is total and conflict-free, m' is also total. By assumption $\mathcal{M}_{\sqsubseteq}$ is preserved by total pushouts, thus μ' is in fact an order morphism. This means that G can be rewritten to some graph larger than S , hence $G \in \text{Pred}(\uparrow S)$. \square

Lemma 3. *It holds that $\uparrow pb_1(S) \supseteq \uparrow \text{Pred}_{\mathcal{Q}}(\uparrow\{S\})$ and $\uparrow pb_2(S) \supseteq \uparrow \text{Pred}(\uparrow\{S\})$.*

Proof. Let G be an element of $\uparrow \text{Pred}(\uparrow\{S\})$. Then there is a minimal representative $G_1 \in \text{Pred}(\uparrow\{S\})$ with $G_1 \sqsubseteq G$ and a rule $r: L \rightarrow R$ rewriting G_1 with a conflict-free match m to some element G_2 of $\uparrow\{S\}$. According to Definition 9 the left diagram below can be extended to the right diagram below.



Since the outer square is a pushout, G_3 is a pushout complement object. Thus, a graph $G_4 \sqsubseteq G_3$ will be obtained by the procedure $pb_2()$ in Step 2 using the rule $\mu_R \circ r$. Summarized, this means that $pb_2()$ computes a graph G_4 for every graph G such that $G_4 \sqsubseteq G_3 \sqsubseteq G_1 \sqsubseteq G$, i.e. every G is represented by an element of $pb_2(S)$.

Now assume $G \in \uparrow \text{Pred}_{\mathcal{Q}}(\uparrow\{S\})$. By definition, the minimal representative G_1 is an element of \mathcal{Q} . We obtain $G_3 \in \mathcal{Q}$, due to the downward closure of \mathcal{Q} . Thus, the procedure $pb_1()$ will compute a graph $G_4 \sqsubseteq G_3$ (with $G_4 \in \mathcal{Q}$), i.e. every G is represented by an element of $pb_1(S)$. \square

Proposition 1. *Let $pb_1()$ and $pb_2()$ be a single backward step of Procedure 1 for Variant 1 and 2 respectively. For each graph S , $pb_1(S)$ is an effective \mathcal{Q} -pred-basis and $pb_2(S)$ is an effective pred-basis.*

Proof. The correctness of $pb_1()$ and $pb_2()$ is a direct consequence of Lemma 2 and 3. Moreover, by the conditions of Procedure 1, the set of minimal pushout complements (possibly restricted to \mathcal{Q}) is finite and computable. Thus, $pb_1()$ and $pb_2()$ are effective. \square

B.3 Subgraph Ordering

Lemma 4. *The subgraph ordering is representable by subgraph morphisms.*

Proof. Let $S \subseteq G$, then by definition S can be obtained from G by a sequence of node and edge deletions of length n . For each G_i and each node or edge $x \in G_i$ we can give a subgraph morphism $\mu_i: G_i \rightarrowtail G_i \setminus \{x\}$, where μ_i is undefined on x and the identity on all other elements. Note that a node can only be deleted if it has no incident edges. Since injectivity and surjectivity are preserved by concatenation, the concatenation $\mu = \mu_1 \circ \dots \circ \mu_n$ is again a subgraph morphism.

Let $\mu: G \rightarrowtail S$ be a subgraph morphism. Since μ is surjective and injective, the inverse of μ is a total, injective morphism $\mu^{-1}: S \rightarrow G$. The image of μ^{-1} is isomorphic to S and a subgraph of G , therefore $S \subseteq G$ holds. \square

Lemma 5. *Subgraph morphisms are preserved by total pushouts.*

Proof. Let $g: G_0 \rightarrow G_2$ be a total morphism and let $\mu: G_0 \rightarrowtail G_1$ be a subgraph morphism, such that μ', g' is the pushout of μ, g as shown in the diagram below.

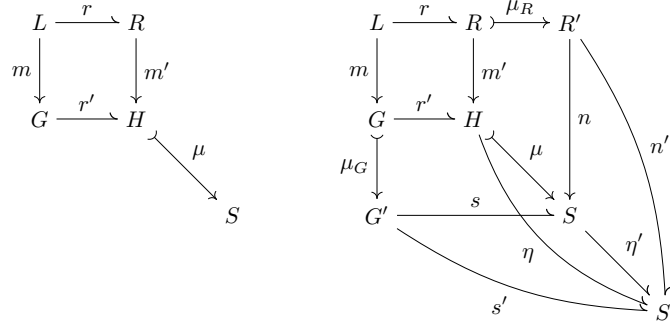
$$\begin{array}{ccc} G_0 & \xrightarrow{\mu} & G_1 \\ \downarrow g & & \downarrow g' \\ G_2 & \xrightarrow{\mu'} & G_3 \end{array}$$

First we show that μ' is injective where it is defined. Assume there are two different elements $x_1, x_2 \in G_2$ such that $\mu'(x_1) = \mu'(x_2)$. For G_3 to be a pushout, both x_1 and x_2 have to have preimages $x'_1, x'_2 \in G_0$ with $g(x'_1) = x_1$ and $g(x'_2) = x_2$. The diagram commutes, thus μ is defined for both elements and these elements are mapped injectively to $x''_1, x''_2 \in G_1$ respectively. Hence, there is a commuting diagram with $g'(x''_1) = \mu'(x_1) \neq \mu'(x_2) = g'(x''_2)$, where there is no mediating morphism from G_3 . Since this violates the pushout properties of the diagram, μ' has to be injective.

It remains to be shown that μ' is surjective. Assume there is an $x_3 \in G_3$ without a preimage under μ' . For the diagram to be a pushout there has to be an $x_1 \in G_1$ with $g'(x_1) = x_3$. Since μ is surjective, there is an $x_0 \in G_0$ with $\mu(x_0) = x_1$ and therefore the diagram does not compute, because $g'(\mu(x_0))$ is defined, but $\mu'(g(x_0))$ is not. Hence, μ' has to be surjective and is a subgraph morphism. \square

Lemma 6. *Subgraph morphisms are pushout closed.*

Proof. Let the morphisms be given as in the left diagram below. We will show the existence of the subgraph morphisms μ_R, μ_G and the morphisms n, s , such that $\mu_G \circ m$ and n are total and $\mu_G \circ m$ is conflict-free wrt. r .



We define $R' = (V_{R'}, E_{R'}, c_{R'}, l_{R'})$ with $V_{R'} = \{v \in V_R \mid \mu(m'(v)) \text{ is defined}\}$, $E_{R'} = \{e \in E_R \mid \mu(m'(e)) \text{ is defined}\}$, $c_{R'}(e) = c_R(e)$ and $l_{R'}(e) = l_R(e)$ for all $e \in E_{R'}$. Note that all vertices of the sequence $c_R(e)$ are in $V_{R'}$ since $\mu \circ m'$ can only be defined for e if it is defined for all attached vertices. On this basis we define $\mu_R(x) = x$ for all $x \in V_{R'} \cup E_{R'}$ (undefined otherwise) and $n = \mu \circ m'$. Obviously, μ_R is injective and surjective where it is defined, hence a subgraph morphism, and n is total, since $\mu \circ m'$ is by definition defined on all elements of R' . Additionally by definition $\mu \circ m' = n \circ \mu_R$, since μ_R is undefined if and only if $\mu \circ m'$ is undefined.

The graph $G' = (V_{G'}, E_{G'}, c_{G'}, l_{G'})$ is defined in a similar way with $V_{G'} = \{v \in V_G \mid \exists v' \in V_L : m(v') = v \vee \mu(r'(v)) \text{ is defined}\}$, $E_{G'} = \{e \in E_G \mid \exists e' \in E_L : m(e') = e \vee \mu(r'(e)) \text{ is defined}\}$, $c_{G'}(e) = c_G(e)$ and $l_{G'}(e) = l_G(e)$ for all $e \in E_{G'}$. Note that $e \in E_{G'}$ again implies that all vertices of $c_G(e)$ are in $V_{G'}$. Also $\mu_G(x) = x$ for all $x \in V_{G'} \cup E_{G'}$ (undefined otherwise) and $s = \mu \circ r'$. Since μ_G is injective and surjective where it is defined, it is a subgraph morphism. Additionally μ_G is defined on all elements of G which have a preimage in L , hence $\mu_G \circ m$ is total and also conflict-free wrt. r , since μ_G is injective and m is conflict-free wrt. r . By definition $\mu \circ r' = s \circ \mu_G$ since $\mu \circ r'$ is undefined on every element of G on which μ_G is undefined.

Finally we show that the outer square is a pushout. We first observe that the outer diagram commutes, since $n \circ \mu_R \circ r = \mu \circ m' \circ r = \mu \circ r' \circ m = s \circ \mu_G \circ m$. Assume there is a graph S' with morphisms $n' : R' \rightarrow S'$ and $s' : G' \rightarrow S'$ and $n' \circ \mu_R \circ r = s' \circ \mu_G \circ m$. The inner square is a pushout, hence there is a morphism $\eta : H \rightarrow S'$ such that $\eta \circ m' = n' \circ \mu_R$ and $\eta \circ r' = s' \circ \mu_G$. Since μ is injective and surjective, the inverse morphism $\mu^{-1} : S \rightarrow H$ is total and well-defined. Thus, there is a unique morphism $\eta' : S \rightarrow S'$ with $\eta' = \eta \circ \mu^{-1}$. Due to the commutativity in the diagram we know that $s' \circ \mu_G = \eta \circ r' = \eta' \circ \mu \circ r' = \eta' \circ s \circ \mu_G$. Since μ_G is surjective, we obtain that $s' = \eta' \circ s$. Analogously we can show that $n' = \eta' \circ n$ commutes and the diagram is a pushout. \square

Lemma 7. *Every GTS satisfies the compatibility condition of Definition 2 with respect to the subgraph ordering.*

Proof. We have to show that whenever $G \Rightarrow H$ and $G \subseteq G'$, then there exists H' with $G' \Rightarrow^* H'$ (here even $G' \Rightarrow H'$) and $G' \subseteq H'$.

Let $m : L \rightarrowtail R$ be a rule and $m : L \rightarrow G$ a matching that is conflict-free wrt. r such that G is rewritten to H , i.e. the upper inner square below is a pushout. Furthermore let $\mu : G' \rightarrowtail G$ be a subgraph morphism, then the inverse morphism μ^{-1} is total and injective. Thus, the morphism $m_\mu : L \rightarrow G'$ with $m_\mu = \mu^{-1} \circ m$ in the diagram below is total and conflict-free wrt. r and G' can be rewritten to H' .

$$\begin{array}{ccccc}
& L & \xrightarrow{r} & R & \\
m_\mu \swarrow & \downarrow m & & \downarrow m' & \searrow m'_\mu \\
& G & \xrightarrow{r'} & H & \\
\mu^{-1} \swarrow & & \xrightarrow{r_\mu} & & \searrow \mu' \\
G' & & & & H'
\end{array}$$

Since the outer square is a pushout, there is a unique morphism $\mu' : H \rightarrow H'$ such that the lower inner square commutes. Furthermore since the upper inner square and the outer square are pushouts, so is the lower inner square. This means that μ' is total and injective since μ^{-1} is and pushouts preserve this properties. Hence, $H \subseteq H'$. \square

Proposition 9. *The subgraph ordering on hypergraphs is a wqo for the set of graphs where the longest undirected path is bounded by a constant.*

Proof. In [6] Ding showed that this proposition holds for undirected, simple graphs with node labels. We will now give an encoding f of hypergraph to such graphs satisfying the following conditions:

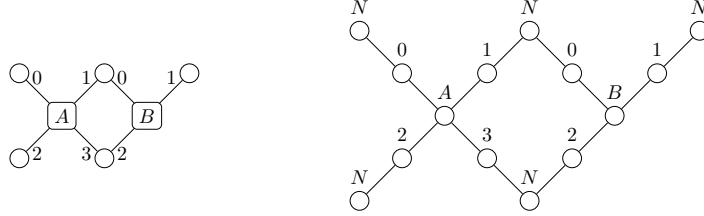
- There is a function $g : \mathbb{N} \rightarrow \mathbb{N}$ such that, if the longest undirected path in a hypergraph G has length k , then the longest undirected path in $f(G)$ has length $g(k)$.
- For every two hypergraphs G_1, G_2 if $f(G_1) \subseteq f(G_2)$ then $G_1 \subseteq G_2$.

If these two properties hold, every infinite sequence G_0, G_1, \dots of hypergraphs with bounded undirected paths can be encoded into an infinite sequence of undirected graphs with bounded paths $f(G_0), f(G_1), \dots$ of which we know that two elements $f(G_i) \subseteq f(G_j)$ exist. Thus, also $G_i \subseteq G_j$ holds.

Let $G = (V, E, c, l)$ be a Λ -hypergraph. We define its encoding as an undirected graph $f(G) = G' = (V', E', l')$ where E' consists of two-element subsets of V' and $l' : V' \rightarrow \Lambda'$ where the components are defined as follows:

$$\begin{aligned}
V' &= V \cup E \cup \{(v, i, e) \mid v \in V, e \in E : c(e) = \alpha v \beta \wedge |\alpha| = i\} \\
E' &= \{\{x, y\} \mid x = (v, i, e) \in V' \wedge (y = v \vee y = e)\} \\
\Lambda' &= \Lambda \cup \{N\} \cup \{n \in \mathbb{N}_0 \mid \exists k \in \Lambda : n < ar(k)\} \\
l'(x) &= \begin{cases} N & \text{if } x \in V \\ l(x) & \text{if } x \in E \\ i & \text{if } x = (v, i, e) \end{cases}
\end{aligned}$$

Note that we assume that $N \notin \Lambda$ and $\Lambda \cap \mathbb{N}_0 = \emptyset$. An example of such an encoding can be seen in the diagram below, where the hypergraph on the left is encoded in the graph on the right-hand side.



We now show that the encoding satisfies the two necessary properties. First we observe, that every (undirected) graph generated by this encoding can be transformed back to a unique hypergraph, up to isomorphism.

Now let G be a hypergraph, where the longest undirected path is bounded by k , we show by contradiction that in $f(G)$ there can not be a path of length at least $4k + 10$. Assume there is such a path in $f(G)$. Apart from the first or last node, all nodes labelled N or $l \in \Lambda$ on this path are adjacent to nodes labelled with $n \in \mathbb{N}_0$ and all nodes labelled with $n \in \mathbb{N}_0$ are adjacent to (exactly) one node labelled with N and one node labelled with $l \in \Lambda$. We now shorten the path in the least possible way to obtain a path of length at least $4k + 4$ which starts and ends with nodes labelled with N . This path can be translated back to a sequence $v_0, e_1, v_1, \dots, v_n, e_{n+1}, v_{n+1}$ since every node labelled with N is a node of (the hypergraph) G and every node labelled with $l \in \Lambda$ is an edge of G . This violates our assumption, that the longest undirected path of G is bounded by k , thus, there is no path of length $4k + 10$ or longer in $f(G)$.

Let G_1, G_2 be hypergraphs such that $f(G_1) \subseteq f(G_2)$. Then there is a total, injective morphism $\mu : f(G_1) \rightarrow f(G_2)$. Since $f(G_i)$ contains (as nodes) all nodes and edges of G_i (for $i \in \{1, 2\}$), μ can be restricted to $V_{G_1} \cup E_{G_1}$ and is then a total, injective morphism $\mu' : G_1 \rightarrow G_2$. The nodes of $f(G_i)$ labelled with natural numbers, ensures the morphism property on the hypergraphs. By inverting μ' we obtain an injective and surjective, but partial morphism from G_2 to G_1 (a subgraph morphism, see Lemma 4), hence $G_1 \subseteq G_2$. \square

Proposition 3 (WSTS wrt. the subgraph ordering). *Let k be a natural number. Every graph transformation system forms a \mathcal{G}_k -restricted WSTS with the subgraph ordering.*

Proof. This is a direct consequence of Lemma 7 and Proposition 9. \square

Proposition 4. *Every \mathcal{G}_k -restricted well-structured GTS with the subgraph order has an effective pred-basis and the (decidability) results of Theorem 1 apply.*

Proof. In Lemma 4, 5 and 6 we have shown that the subgraph ordering satisfies the conditions of Procedure 1. Furthermore, the set of minimal pushout complements – not just restricted to \mathcal{G}_k – can be computed in the same way as it is done in [12] for the minor ordering, such that both variants of Procedure 1 are applicable. \square

Proposition 5. *Let $k > 2$ be a natural number. The restricted coverability problem for \mathcal{G}_k -restricted well-structured GTS with the subgraph ordering is undecidable.*

Proof. We reduce the control state reachability problem of Minsky machines to the restricted coverability problem using the subgraph ordering on the set of graphs \mathcal{G}_2 , where the length of the longest undirected path is less than or equal to two. Let $(Q, \Delta, (q_0, m, n))$ be the Minsky machine. We define a GTS using $\{q, q^B \mid q \in Q\} \cup \{c_1, c_2, X\}$ as the set of labels. The initial graph is shown in Figure 4 and illustrates how configurations of the Minsky machine are represented as graphs.

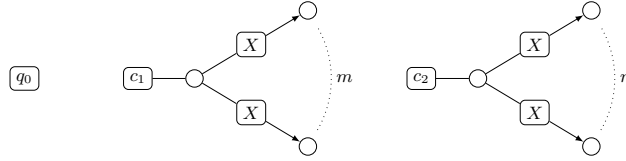


Fig. 4: The initial configuration of the Minsky machine represented by a graph

For each transition rule of the Minsky machine, we add a graph transformation rule as shown in Figure 5. A counter is represented as a star-like structure with the counter's main node as centre, where the value of the counter is the number of attached X -edges. Incrementing and decrementing corresponds to creating and deleting X -edges. Regardless of the counter's value, the longest undirected path of this structure has at most length two.

The zero-test adds two X -edges and blocks the state-edge, such that the rewritten graph has a undirected path of length three if and only if the counter was not zero (i.e. had an X -edge attached). The auxiliary rules unblock the state to enable further computation.

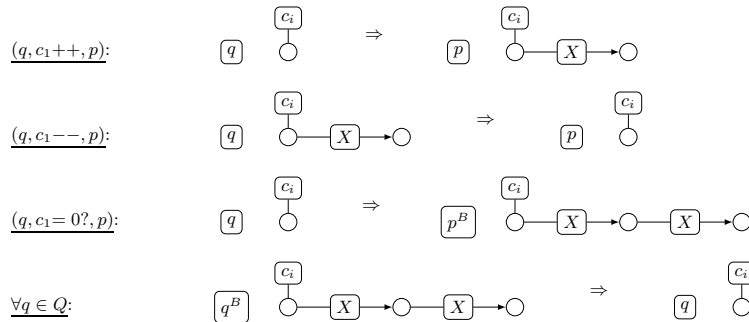


Fig. 5: Translation of Minsky rules to GTS rules

Obviously, if there is a sequence of transitions of the Minsky machine which leads from a configuration (q_0, m, n) to a state q_f , this sequence can be copied in the GTS and every graph generated through this sequence is in \mathcal{G}_2 . On the other hand, if the graph consisting of a single q_f -edge is \mathcal{G}_2 -restricted coverable in the GTS, there is a sequence of rule applications corresponding to a sequence of transitions of the Minsky machine. Since this rule applications generate only graphs in \mathcal{G}_2 , the zero-test-rule is only applied if the counters value is in fact zero and the sequence of transitions is valid.

Instead of adding and removing a path of length two in the last two rules of Figure 5 one can add and remove a path of length k to show the undecidability for \mathcal{G}_k -restricted well-structured GTS. \square

B.4 Induced Subgraph Ordering

Lemma 8. *The induced subgraph ordering is presentable by induced subgraph morphisms.*

Proof. Let $\mu_1 : G_1 \triangleright \rightarrow G_2$ and $\mu_2 : G_2 \triangleright \rightarrow G_3$ be two induced subgraph morphisms. Induced subgraph morphisms are closed under composition, since injectivity and surjectivity are preserved and if $\mu_2 \circ \mu_1$ is undefined for some edge e , then μ_1 is undefined on e or μ_2 is undefined on $\mu_1(e)$ implying that $\mu_2 \circ \mu_1$ is undefined for at least one node of e .

For some graph G we can obtain any induced subgraph G' by a sequence of node deletions including all attached edges. Each morphisms $\mu_i : G_i \triangleright \rightarrow G_{i+1}^x$ of this sequence, where G_{i+1}^x is obtained by deleting the node x and all its attached edges from G_i , is an induced subgraph morphisms and since they are closed under composition, the entire sequence is as well.

On the other hand every induced subgraph morphism $\mu : G \triangleright \rightarrow G'$ can be split into a sequence of node deletions (deleting all attached edges), since every deleted edge is attached to a deleted node, hence $G' \trianglelefteq G$. \square

Lemma 9. *Induced subgraph morphisms are preserved by total pushouts.*

Proof. Since every induced subgraph morphism is also a subgraph morphism, μ' is injective and surjective by Lemma 5. Let $e \in E_{G_2}$ be an edge on which μ' is undefined. e has a preimage $e' \in G_0$ since otherwise the pushout of μ and g would contain e . Since $\mu'(g(e'))$ is undefined, so is $g'(\mu(e'))$. In fact μ is undefined for e' because otherwise g and μ would be defined on e and e would be in the pushout. μ is an induced subgraph morphism, thus at least one of the nodes v attached to e' is undefined and also μ' has to be undefined on $g(v)$ for the diagram to commute. \square

Lemma 10. *Induced subgraph morphisms are pushout closed.*

Proof. In Lemma 6 we have shown that there are subgraph morphisms μ_R and μ_G if μ is a subgraph morphism. We will show that these morphisms are induced subgraph morphisms if μ is an induced subgraph morphism.

Let $e \in E_R$ be an edge on which μ_R is undefined. By definition $\mu(m'(e))$ is undefined and since m' is total, μ is undefined on $m'(e)$ (which is defined). Hence, at least one node v attached to $m'(e)$ has no image under μ and all its preimages under m' (which exist since e has a preimage) are undefined under μ_R . Thus, e is attached to at least one node on which μ_R is undefined on.

Let $e' \in E_G$ be an edge on which μ_G is undefined. By definition $\mu(r'(e'))$ is undefined and e' has no preimage under m . Because of the latter property, e' is in the pushout H and therefore defined under r' . Thus, μ is undefined on $r'(e)$ and on at least one attached node. All preimages under r' of this node are undefined under μ_G since the diagram commutes. Hence, μ_G is an induced subgraph morphism. \square

Lemma 11. *Every GTS satisfies the compatibility condition of Definition 2 with respect to the induced subgraph ordering.*

Proof. We modify the proof of Lemma 7 by additionally showing that the reverse of μ' is an induced subgraph morphism. Assume there is an edge $e \in E_{H'}$, where all attached nodes have a preimage under μ' but e has none. Since $r', \mu^{-1}, \mu', r_\mu$ is a pushout, this can only be the case if all nodes attached to e have a preimage in G and G' and e has a preimage in G' . Because μ is an induced subgraph morphism, e has a preimage in G . Due to commutativity r' cannot be undefined on this preimage, thus, e has to have a preimage in H , violating the assumption. \square

To prove Proposition 10 we adapt Dings proof using the notion of type of a graph.

Definition 14 (Type of a Graph). *A graph which consists of at most a single node with possibly attached edges has type one. A connected graph containing at least two nodes has at most type n , if there is a node v so that the deletion of v and all attached edges splits the graph into components which each have the type $n - 1$. The type of a non-connected graph is the maximal type of its components.*

Lemma 12 ([6]). *Every directed graph, where the longest undirected path has length n , has at most type $n + 2$.*

Note that contrary to Ding our type is bounded by $n + 2$ instead of n , because we measure path lengths via the number of edges instead of nodes and Ding excludes paths of length n to obtain graphs of type at most n .

Proposition 10. *Let n, k be natural numbers. The induced subgraph ordering is a wqo on the set of directed, edge-labelled graphs, where the longest undirected path is bounded by n and every two nodes are connected by at most k parallel edges with the same label (bounded edge multiplicity).*

Proof. We prove this proposition by induction over the type of a graph, adapting Dings proof in [6] that undirected, node-labelled graphs of bounded type are well-quasi-ordered by the induced subgraph order. Because of Lemma 12 we know

that the result for bounded types automatically transfers to bounded undirected paths. To prove this proposition we use hypergraphs which are additionally node labelled, i.e. there is a second alphabet Σ of node labels and a (total) labelling function $\sigma : V_G \rightarrow \Sigma$. We obtain classical directed graphs if $|\Sigma| = 1$.

Let G_1, G_2, \dots be an infinite sequence of graphs of type n and with edge multiplicity bounded by k . If $n = 1$ then every G_i consists of a single node with up to $k \cdot |\Lambda|$ attached loops. Since the sets of node and edge labels are finite, there are only finitely many possibilities to attach up to $k \cdot |\Lambda|$ edges to the node, thus $G_i \sqsubseteq G_j$ for some $i < j$, i.e. \sqsubseteq is a wqo on the set of all such graphs.

Now let $n > 1$. Then there is a node $v_i \in G_i$ such that the deletion of v_i (and its attached edges) splits the graph into components $G_{i,q}$ (for $1 \leq q \leq \ell_i$) of type at most $n - 1$. We define \tilde{G}_i to be the graph containing only v_i and its attached loops. Additionally we define $\hat{G}_{i,q}$ to be $G_{i,q}$ where the label $\sigma(y)$ of every node y is changed to $\sigma'(y) = (f_y, \sigma(y))$, where $f_y : \Lambda \rightarrow \{0, 1, \dots, k\}^2$ is a function such that $f_y(\lambda) = (a, b)$ where a is the number of incoming and b of outgoing λ -labelled edges attached to both y and v_i . Since there are only finitely many possible functions f_y (due to the multiplicity constraint), the set of labels remains finite. We extend \sqsubseteq to sequences such that $(\tilde{G}_i, \hat{G}_{i,1}, \dots, \hat{G}_{i,\ell_i}) \sqsubseteq^* (\tilde{G}_j, \hat{G}_{j,1}, \dots, \hat{G}_{j,\ell_j})$ if and only if $\tilde{G}_i \sqsubseteq \tilde{G}_j$ and there are p_1, \dots, p_{ℓ_i} with $1 \leq p_1 < \dots < p_{\ell_i} \leq \ell_j$ such that $\hat{G}_{i,q} \sqsubseteq \hat{G}_{j,p_q}$. As shown for the case $n = 1$, \sqsubseteq is a wqo on all \tilde{G}_i and since the graphs $\hat{G}_{i,q}, \hat{G}_{j,p_q}$ are of type $n - 1$, they are well-quasi-ordered by induction hypothesis. Hence, due to Higman [11] \sqsubseteq^* is also a wqo and there are indices $i < j$ such that $(\tilde{G}_i, \hat{G}_{i,1}, \dots, \hat{G}_{i,\ell_i}) \sqsubseteq^* (\tilde{G}_j, \hat{G}_{j,1}, \dots, \hat{G}_{j,\ell_j})$. It remains to be shown that this implies $G_i \sqsubseteq G_j$. By Lemma 8 there are induced subgraph morphisms $\mu_0 : \tilde{G}_j \rightarrow \tilde{G}_i$ and $\mu_q : \hat{G}_{j,p_q} \rightarrow \hat{G}_{i,q}$ for $1 \leq q \leq \ell_i$. We define the morphism $\mu : G_j \rightarrow G_i$ as

$$\mu(x) = \begin{cases} v_i & \text{if } x = v_j \\ \mu_q(x) & \text{if } x \in \hat{G}_{j,p_q} \text{ for some } q \\ \mu_0(x) & \text{if } x \in E_{\tilde{G}_j} \text{ and } c_{\tilde{G}_j}(x) = v_j v_j \\ \mu^v(x) & \text{if } x \in E_{G_j} \text{ and } c_{G_j}(x) = v_j v \vee c_{G_j}(x) = v v_j \\ & \text{for } v_j \neq v \in V_{G_j} \text{ and } \mu(v) \text{ is defined} \\ \text{undefined} & \text{else} \end{cases}$$

where μ^v is any total, bijective morphism from G_j restricted to v_j, v and the edges between them to G_i restricted to $v_i, \mu_q(v)$ if $v \in \hat{G}_{j,p_q}$ and any edges between them (both not including loops). Note that μ^v exists since v and $\mu_q(v)$ are labelled with some (f, α) , thus the number of edges between v_j and v is equal to the number of edges between v_i and $\mu_q(v)$ for all labels and directions.

We now show that μ is an induced subgraph morphism. First note that μ is a valid morphism since μ_q, μ_0 and μ^v are morphisms and labels of edges in G_i, G_j are the same as their representative in $\hat{G}_{j,p_q}, \hat{G}_{i,q}$ and representatives of nodes are labelled with (f, α) while the origin is labelled α also implying equality on labels. We then observe that μ is injective and surjective, since μ_q, μ_0 and μ^v

are all injective and surjective and v_j is mapped to v_i . Assume there is an edge $e \in E_{G_j}$ for which μ is undefined. If e is contained in one of the components \widehat{G}_{j,p_q} or in \widetilde{G}_j , at least one attached node is undefined, since μ_q and μ_0 are an induced subgraph morphisms. If e connects v_j and a node v of a component $\widehat{G}_{j,z}$, then either z is not of the form p_q and μ is undefined on $\widehat{G}_{j,z}$ or $z = p_q$ and $\mu(v)$ is undefined since otherwise μ^v has a mapping for e . Since μ is an induced subgraph morphism, we obtain that $G_i \leq G_j$. \square

Proposition 6 (WSTS wrt. the induced subgraph ordering). *Let n, k be natural numbers and let $\mathcal{G}_{n,k}$ be a set of directed, edge-labelled graphs, where the longest undirected path is bounded by n and every two nodes are connected by at most k parallel edges with the same label (bounded edge multiplicity). Every GTS forms a $\mathcal{G}_{n,k}$ -restricted WSTS with the induced subgraph ordering.*

Proof. This is a direct consequence of Lemma 11 and Proposition 10. \square

Proposition 7. *Every $\mathcal{G}_{n,k}$ -restricted well-structured GTS with the induced subgraph order has an effective $\mathcal{G}_{n,k}$ -pred-basis and the (decidability) results of Theorem 1 apply.*

Proof. As shown in Lemma 8, 9 and 10 the induced subgraph ordering satisfies the conditions of Procedure 1. The computation of minimal pushout complements is more involved than in the subgraph case. This is due to the fact that if a rule deletes a node, all attached edges are deleted, even if these edges have no preimage in L . Adding an edge to a pushout complement and attaching it to a node which is deleted by the rule, results in another pushout complement. Contrary to the subgraph ordering these pushout complements are not already represented by the graph without the edge, if the induced subgraph ordering is used, but we can compute them as follows:

1. Let $(r : L \rightarrow R) \in \mathcal{R}'$ be a rule and $m : R \rightarrow G$ a match calculated in Step 1 of Procedure 1. Calculate the set of minimal pushout complements \mathcal{G}_{poc} wrt. the subgraph ordering restricted to $\mathcal{G}_{n,k}$.
2. For all pushout complement objects $X \in \mathcal{G}_{poc}$ with morphism $r' : X \rightarrow G$, add all X' to \mathcal{G}_{poc} , where X' can be obtained by adding an edge to X , which is attached to at least one node on which r' is undefined. Do not add X' if it exceeds the bounded multiplicity.
3. Perform Step 2 until \mathcal{G}_{poc} becomes stationary, which will be the case since the multiplicity is bounded. The set \mathcal{G}_{poc} is then the set of minimal pushout complement objects wrt. the induced subgraph ordering.

\square